LA-UR-01-2674
*Approved for public release;
distribution is unlimited.*

Title:

Fully Consistent Diffusion Synthetic Acceleration of Linear Discontinuous Transport Discretizations on Three-Dimensional Unstructured Meshes

Author(s): James S. Warsa, Todd A. Wareing and Jim E. Morel

Submitted to:

FULLY CONSISTENT DIFFUSION SYNTHETIC ACCELERATION OF LINEAR DISCONTINUOUS TRANSPORT DISCRETIZATIONS ON THREE–DIMENSIONAL UNSTRUCTURED MESHES

James. S. Warsa, Todd A. Wareing and Jim E. Morel
Transport Methods Group
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Send proofs to:

James S. Warsa
Los Alamos National Laboratory
CCS–4, MS D409
Los Alamos, NM 87545

E-mail: *warsa@lanl.gov*
FAX: 505-665-5538

Number of pages: 31
Number of figures: 4
Number of tables: 3

FULLY CONSISTENT DIFFUSION SYNTHETIC ACCELERATION OF LINEAR DISCONTINUOUS TRANSPORT DISCRETIZATIONS ON THREE–DIMENSIONAL UNSTRUCTURED MESHES

James S. Warsa, Todd A. Wareing and Jim E. Morel
Transport Methods Group
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Abstract

We recently presented a method for efficiently solving linear discontinuous discretizations of the P_1 equations in two–dimensional Cartesian geometry. The linear system was efficiently solved with Krylov iterative methods and a novel two–level preconditioner based on a continuous discretization of the diffusion equation. Here we extend the preconditioned solution method to three–dimensional, unstructured tetrahedral meshes. Solution of the P_1 equations forms the basis of a diffusion synthetic acceleration (DSA) scheme for three–dimensional S_N transport calculations. The P_1 equations and the transport equation are both discretized with isoparametric linear discontinuous finite elements so that the DSA method is fully consistent. Fourier analysis in three dimensions and computational results show that this DSA scheme is stable and very effective. The fully consistent method is compared to other "partially consistent" DSA schemes. Results show that the effectiveness of the partially consistent schemes can degrade for skewed or optically thick mesh cells. In fact, one such scheme can degrade to the extent of being unstable even though it is both unconditionally stable and effective on rectangular grids. Results for a model application show that our fully consistent DSA method can outperform the partially consistent DSA schemes under certain circumstances.

1. INTRODUCTION

In this paper, we consider diffusion synthetic acceleration schemes for the linear discontinuous finite element method (DFEM) discretization of the S_N transport equation on three–dimensional unstructured tetrahedral meshes. See Ref. 1 and references therein for a description of the DFEM for the S_N equations on unstructured meshes.

For many problems diffusion synthetic acceleration (DSA) is needed to accelerate convergence of the S_N source iterations.[2,3] It is well known that the discretization of the diffusion equation in a DSA scheme must be "consistent" with the S_N transport equations discretization in order to be effective and robust for a wide range of problems.[4,5] The effectiveness of a DSA scheme can be measured by the spectral radius of the accelerated algorithm. The spectral radius is always less than one for a useful algorithm. The closer it is to zero the more quickly the source iterations converge. Without spatial discretization, the DSA algorithm drastically reduces the spectral radius in highly diffusive problems whose spectral radius would otherwise be close to one.[4,6] In the spatially discretized case, consistency is needed for DSA to achieve this level of effectiveness.

Our fully consistent DSA method is based on a discretization of the P_1 equations using the same linear finite element basis used in the linear DFEM discretization of the S_N equations on tetrahedral meshes. In both discretizations, discontinuities are introduced by "upwinding" the variables of interest. The linear system of the P_1 discretization is large and sparse. This is because there is a large number (sixteen) of unknowns per cell making direct solution methods impractical even for problems of intermediate size. We therefore solve the discontinuous P_1 equations iteratively with Krylov–subspace methods. Although we only have to store the nonzeros of the sparse matrix the linear system will require significant memory and matrix vector products can be expensive.

An alternative to consistent DSA schemes are those in which the diffusion equation discretizations are only partially consistent with the transport discretization.[1,7–9] The idea is to reduce the complexity and increase the efficiency of the DSA algorithm. In one case, a scheme with a symmetric positive definite (SPD) continuous discretization of the diffusion equation centered on the mesh nodes is combined with a prescription to compute (or "update") the necessary discontinuous quantities on a local, cell–by–cell basis using the continuous diffusion equation solution.[1,7] We will refer to this method as the WLA (Wareing, Larsen and Adams) DSA scheme. The fact that the discretization is centered on the nodes means the linear system can be relatively small because often there are many fewer nodes (vertices) than cells on unstructured tetrahedral meshes. Because this system is SPD it can be solved by the method of conjugate gradients, which has modest memory requirements and low computational complexity. Thus, the WLA DSA method can be very efficient. In another approach, Larsen's so–called four step method[4] is modified to find a nonsymmetric linear system for the discontinuous scalar fluxes.[8,9] This is referred to as modified four step

(M4S) DSA. Because this method involves only the scalar fluxes, the dimension of the linear system is one quarter of the dimension of the fully consistent method. Even with the smaller dimension, direct methods can be impractical for large problems. However, a Krylov–subspace iterative method will need less work per iteration than the fully consistent method and the partially consistent diffusion equations can be solved more efficiently.

The increased efficiency of the partially consistent DSA methods can sometimes come at the cost of decreased effectiveness. This can be tolerated as long as the cost of a DSA step is cheap. However, partial consistency could result in situations where the DSA method no longer accelerates the transport iterations. We found that the effectiveness of the WLA DSA scheme decreases as the cells in a problem become optically thick and diffusive. We also discovered that in certain problems with skewed cells, the M4S method degrades to the point where it causes the S_N source iterations to diverge.

Because the DSA scheme we present here is consistent, the spectral radius and, hence, the number of source iterations will be reduced significantly. But it can be competitive with the partially consistent methods only if the solution of the discontinuous P_1 equations can be computed efficiently. We are using a Krylov–subspace iterative solution so we can improve solution efficiency with a good preconditioner. In Ref. 10 we presented a two–level preconditioner for the discontinuous P_1 equations in two–dimensional Cartesian geometry. Analysis and numerical experimentation showed that this preconditioner was efficient and very effective. We have extended this two–level preconditioned solution technique to the discontinuous P_1 equations on three–dimensional unstructured tetrahedral meshes. The solution method was implemented in the S_N transport code ATTILAV2[11] as a fully consistent DSA scheme. Our purpose is to discuss the linear DFEM discretization and the two–level preconditioned iterative solution of the P_1 equations. We also investigate the overall efficiency of the accelerated S_N transport solutions, comparing the fully consistent DSA method to the partially consistent methods.

The remainder of the paper is organized as follows. In the next section we derive a linear DFEM discretization of the P_1 equations on tetrahedral meshes. We then discuss how we solve these equations with a Krylov–subspace iterative method and our two–level preconditioner. The third section presents the details of three–dimensional Fourier analysis on tetrahedra that we use to predict the effectiveness of a DSA scheme. In the fourth section we compare Fourier analysis predictions of the spectral radius to actual measurements of the spectral radius made with ATTILAV2. A moderately sized, realistic example problem is used to measure the computational effort of the accelerated S_N solution methods. All the results compare the fully consistent DSA method to the partially consistent DSA schemes which have also been implemented in ATTILAV2. The paper concludes with some summary closing remarks.

2. DISCONTINUOUS P_1 EQUATIONS ON TETRAHEDRAL MESHES

In this section, we will derive a DFEM discretization for the diffusion equation by noting that the (time–independent) system of coupled, first order P_1 equations and the second order diffusion equation are equivalent. We can then discretize the first order system with a linear DFEM similar to that used for the S_N transport equation. The difference is that the angular flux unknown associated with the transport operator is naturally identified with a unique direction of particle flow. These flow directions facilitate the introduction of discontinuities into the discretization. In contrast, quantities corresponding to particle flows do not exist naturally. Instead, the particle flows are *defined* using one of several possible approaches.[8–10, 12]

There has recently been a great deal of interest and activity in the applied mathematics community relating to the development of discontinuous Galerkin methods for the discretization of partial differential equations.[13] To connect with this literature, we point out that the linear DFEM discretization of the S_N transport equation can be viewed as a linear discontinuous, Galerkin finite element method, denoted by dG(1). Similarly, our consistent DFEM discretization of the P_1 equations can be viewed as a "mixed" dG(1) method for the diffusion equation.[12] The numerical analysis of mixed discontinuous methods for elliptic operators, like the diffusion equation, is an active area of research.

This section begins by describing the discontinuous DFEM discretization of the P_1 equations on tetrahedra. This is followed by a description of the two–level preconditioning technique using a linear continuous discretization of the diffusion equation. A brief discussion of the DSA algorithm follows that, and the section concludes with ways to improve the efficiency of the DSA algorithm.

2.1 The Discretized Equations

The steady–state P_1 equations in three–dimensional geometry are

$$\nabla \cdot \boldsymbol{J}(\boldsymbol{r}) + \sigma_a(\boldsymbol{r})\, \Phi(\boldsymbol{r}) = Q_0(\boldsymbol{r}) \tag{1a}$$

$$\frac{1}{3}\nabla \Phi(\boldsymbol{r}) + \sigma_t(\boldsymbol{r})\, \boldsymbol{J}(\boldsymbol{r}) = \boldsymbol{Q}_1(\boldsymbol{r}). \tag{1b}$$

The usual particle transport notation is used here: $\Phi(\boldsymbol{r})$ represents the scalar flux (zeroth angular moment of the angular flux), $\boldsymbol{J}(\boldsymbol{r})$ represents the current vector (first angular moment of the angular flux), and $\boldsymbol{r} \in V$ is the position vector in some domain V. The source terms $Q_0(\boldsymbol{r})$ and $\boldsymbol{Q}_1(\boldsymbol{r})$ are the zeroth and first angular moments of an inhomogeneous source, a material property. It is often assumed the material emits particles isotropically such that $\boldsymbol{Q}_1(\boldsymbol{r})$ is zero. The first expression is called the balance equation. The second is a vector equation referred to as the first moment equation(s). For clarity they will consistently be written in this order.

Boundary conditions for the P_1 equations are specified by separating the flow of particles through a surface into inwardly and outwardly directed flows. One way to do this is to use the P_1 approximation itself and assume the angular flux is linear in angle: $\Psi(\boldsymbol{r},\hat{\Omega}) = \frac{1}{4\pi}\Phi(\boldsymbol{r}) + \frac{3}{4\pi}\hat{\Omega}\cdot\boldsymbol{J}(\boldsymbol{r})$. Under this approximation, the inwardly directed flow (partial current) of particles through a surface located at $\boldsymbol{r}_s$ can be expressed as

$$J^{in}(\boldsymbol{r}_s) = \int_{\left(\hat{n}\cdot\hat{\Omega}\right)<0} \left(\hat{n}\cdot\hat{\Omega}\right)\,\Psi(\boldsymbol{r}_s,\hat{\Omega}) = \frac{1}{4}\Phi(\boldsymbol{r}_s) - \frac{1}{2}\hat{n}\cdot\boldsymbol{J}(\boldsymbol{r}_s), \tag{2a}$$

and an outwardly directed flow as

$$J^{out}(\boldsymbol{r}_s) = \int_{\left(\hat{n}\cdot\hat{\Omega}\right)>0} \left(\hat{n}\cdot\hat{\Omega}\right)\,\Psi(\boldsymbol{r}_s,\hat{\Omega}) = \frac{1}{4}\Phi(\boldsymbol{r}_s) + \frac{1}{2}\hat{n}\cdot\boldsymbol{J}(\boldsymbol{r}_s). \tag{2b}$$

We will consider vacuum or reflection boundary conditions. For vacuum conditions, assume no external angular flux of particles enters the through the external boundary surface, $\partial V \subset V$, that is, $\Psi(\boldsymbol{r_s},\hat{\Omega}) = 0$ for $\boldsymbol{r}_s \in \partial V$ and $\left(\hat{n}\cdot\hat{\Omega}\right) < 0$. Then the vacuum condition, or Marshak boundary condition, relates the scalar flux and currents on the boundary through Eq. 2a by setting $J^{in}(\boldsymbol{r}_s) = 0$ for $\boldsymbol{r}_s \in \partial V$. Reflection boundary conditions are specified by simply setting $J^{in}(\boldsymbol{r}_s) = J^{out}(\boldsymbol{r}_s)$. We will also need Eqs. 2a and 2b later when we "upwind" our discontinuous discretization.

We can construct the DFEM discretization of the P_1 equations by defining a linear finite element basis on a tetrahedral cell $T_k \in V$. A local ordering of the faces and vertices of a cell is established such that the face i is opposite the vertex i. The bases $L_i, i = 1,4$ are then defined in terms of the "local" linear barycentric coordinates

$$L_1 = x, \tag{3a}$$

$$L_2 = y, \tag{3b}$$

$$L_3 = z, \tag{3c}$$

$$L_4 = 1 - L_1 - L_2 - L_3 = 1 - x - y - z, \tag{3d}$$

on the "master" tetrahedron as illustrated in Fig. 1. The basis functions are unity at their respective vertices and are zero at the other three vertices. For the isoparametric elements we use here, the linear basis functions *are* the barycentric coordinates. Equations 3 establish a mapping between the global Cartesian coordinates of the mesh vertices and the local barycentric coordinates. Their use simplifies the derivation and is easily generalized to higher order elements. More on the use of barycentric coordinates can be found in Refs. 14 and 1.

Now we expand the scalar fluxes an currents on a cell in terms of the basis functions. These approximations are respectively denoted by Φ_h and $\boldsymbol{J}_h$ and the general discrete problem is as follows.

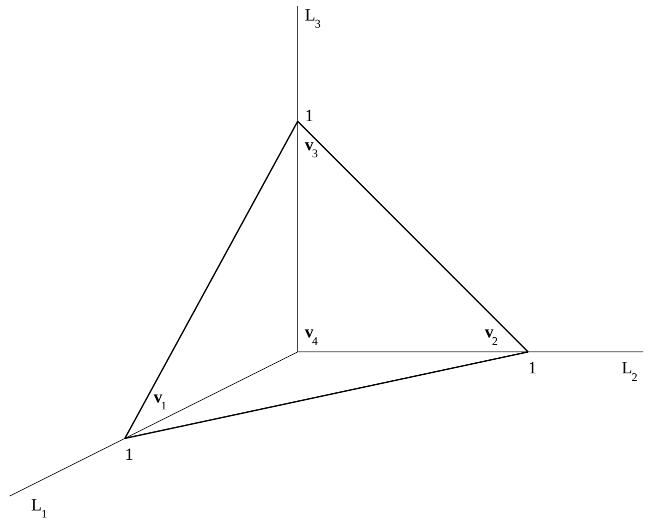

Figure 1. The master tetrahedron with vertices in (x, y, z) coordinates at $\boldsymbol{v}_1 = (0,0,0)$, $\boldsymbol{v}_2 = (1,0,0)$, $\boldsymbol{v}_3 = (0,0,1)$, and $\boldsymbol{v}_4 = (0,1,0)$. The local coordinate axes are the barycentric coordinates $L_1 = x$, $L_2 = y$ and $L_3 = z$ (the fourth barycentric coordinate is not shown).

Compute the approximations Φ_h and $\boldsymbol{J}_h$ on a cell T_k satisfying

$$\int_{\partial T_k} \left(\hat{n} \cdot \boldsymbol{J}_h^b\right) u_h \, dS - \int_{T_k} \boldsymbol{J}_h \cdot \boldsymbol{\nabla} u_h \, dV + \sigma_{a,k} \int_{T_k} \Phi_h u_h \, dV = \int_{T_k} Q_0 u_h \, dV, \tag{4a}$$

$$\frac{1}{3} \int_{\partial T_k} \Phi_h^b \left(\hat{n} \cdot \boldsymbol{w}_h\right) dS - \frac{1}{3} \int_{T_k} \Phi_h \left(\boldsymbol{\nabla} \cdot \boldsymbol{w}_h\right) dV + \sigma_{t,k} \int_{T_k} \boldsymbol{J}_h \cdot \boldsymbol{w}_h \, dV = \int_{T_k} \boldsymbol{Q}_1 \cdot \boldsymbol{w}_h \, dV, \tag{4b}$$

for all trial functions u_h and $\boldsymbol{w}_h$.

We have used the divergence theorem to integrate terms involving gradients. This is necessary for introducing the discontinuous approximation.

The linear trial space consists of the four scalar trial functions u_h, which are just the four basis function L_i, and the four vector trial functions $\boldsymbol{w}_h$, which are written in terms of the basis functions as $L_i\hat{\imath} + L_i\hat{\jmath} + L_i\hat{k}$ for $i = 1, \ldots, 4$. Equations 4 are written at the vertex of every cell and Eq. 4b is written separately for each of the x, y, and z components at the four cell vertices. The result is sixteen equations in sixteen unknowns for every cell in the mesh. The unknowns are the values of flux, $\Phi_{i,k}$ and three current components J_i^x, J_i^y, and J_i^z at the four cell vertices, $i = 1, 4$. The flux and currents are discontinuous, defined separately in each cell as the limiting values of $\Phi(\boldsymbol{r})$ and $\boldsymbol{J}(\boldsymbol{r})$ as $\boldsymbol{r} \to \boldsymbol{v}_i$ from within the cell.

The integrals over the boundary of the tetrahedral cells, ∂T_k, contain the quantities Φ_h^b and $\boldsymbol{J}_h^b$, indicating that they are "boundary" terms. They are uniquely defined in terms of the particle flows through the cell faces, using a linear combination of the discrete discontinuous unknowns from the two adjacent cells sharing a particular face. The three discontinuous values of the scalar flux and currents on the cell face of a particular cell are used to define the outgoing particle flow through the face. The three discontinuous values of the flux and currents from the adjacent cell that shares the face are used to define the incoming particle flow through the face. If the face of a cell constitutes an external boundary, the incoming flow is specified according to the boundary conditions.

The result of the integrations over the cell boundaries in Eqs. 4 involve "boundary" terms of the form $\boldsymbol{a}_j \Phi_i^b$ and $\left(\boldsymbol{a}_j \cdot \boldsymbol{J}_i^b\right)$ at vertex i. The "area vector" is defined as $\boldsymbol{a}_j = \hat{n}_j a_j$ for a face j whose area is a_j with outward normal $\hat{n}_j$. One way to define the upwind particle flows for the discontinuous discretization is to use discrete versions of the partial currents as follows. First, adding and subtracting the continuous partial currents we can write

$$\Phi_h^b = 2\left(J^{out} + J^{in}\right) \tag{5a}$$

$$\left(\hat{n} \cdot J_h^b\right) = J^{out} - J^{in}. \tag{5b}$$

Use these relationships for vertex i and specify boundary conditions by setting ξ_j according to

$$\xi_j = \begin{cases} 0 & \text{if face } j \text{ is vacuum or internal,} \\ 1 & \text{if face } j \text{ is reflective,} \end{cases}$$

to get the necessary quantities:

$$\boldsymbol{a}_j \, \Phi_i^b = 2 \, \boldsymbol{a}_j \left(J_{j,i}^{out} \, [1 + \xi_j] + J_{j,i}^{in} \, [1 - \xi_j]\right) \tag{6a}$$

$$\left(\boldsymbol{a}_j \cdot J_i^b\right) = |\boldsymbol{a}_j| \left(J_{j,i}^{out} - J_{j,i}^{in}\right) [1 - \xi_j]. \tag{6b}$$

The particle flow definitions are completed by defining the partial currents in these last expressions. The flows into and out of face j at vertex i are

$$J_{j,i}^{in} = \frac{1}{4} \, \Phi_i^{ext} - \frac{1}{2} \hat{n}_j \cdot \boldsymbol{J}_i^{ext} \tag{6c}$$

$$J_{j,i}^{out} = \frac{1}{4} \, \Phi_i + \frac{1}{2} \hat{n}_j \cdot \boldsymbol{J}_i, \tag{6d}$$

where "*ext*" denotes exterior quantities. They are the quantities from the adjacent cell that shares face j with cell k and "across" the face from vertex i.

Another way to define the upwinded boundary terms is borrowed from computational fluid dynamics.[15,16] Consider the homogeneous, time–dependent P$_1$ equations in a void,

$$\frac{\partial \Phi(\boldsymbol{r}, t)}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{J}(\boldsymbol{r}, t) = 0, \tag{7a}$$

$$\frac{\partial \boldsymbol{J}(\boldsymbol{r}, t)}{\partial t} + \frac{1}{3} \boldsymbol{\nabla} \Phi(\boldsymbol{r}, t) = 0, \tag{7b}$$

and the advection of a plane wave oriented in the normalized direction $\hat{n} = n_x\hat{\imath} + n_y\hat{\jmath} + n_z\hat{k}$,

$$\boldsymbol{u} = \begin{bmatrix} \Phi(\boldsymbol{r},t) \\ J^x(\boldsymbol{r},t) \\ J^y(\boldsymbol{r},t) \\ J^{z(\boldsymbol{r},t)} \end{bmatrix} = \begin{bmatrix} \alpha_0(t) \\ \alpha_x(t) \\ \alpha_y(t) \\ \alpha_z(t) \end{bmatrix} e^{i\kappa(\hat{n}\cdot\boldsymbol{r})} \equiv \boldsymbol{\alpha} e^{i\kappa(\hat{n}\cdot\boldsymbol{r})}. \tag{8}$$

Eqs. 7 can then be written as a first–order vector wave equation

$$\boldsymbol{\alpha}_t + i\kappa\boldsymbol{H}\boldsymbol{\alpha} = 0, \tag{9}$$

the subscript t denoting a time derivative. The matrix

$$\boldsymbol{H} = \begin{bmatrix} 0 & n_x & n_y & n_z \\ \frac{1}{3}n_x & 0 & 0 & 0 \\ \frac{1}{3}n_y & 0 & 0 & 0 \\ \frac{1}{3}n_z & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

is then diagonalized such that $\boldsymbol{H} = \boldsymbol{R}\boldsymbol{\Lambda}\boldsymbol{R}^{-1}$, where

$$\boldsymbol{R} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ \frac{1}{\sqrt{3}}n_x & -\frac{1}{\sqrt{3}}n_x & 1 & 0 \\ \frac{1}{\sqrt{3}}n_y & -\frac{1}{\sqrt{3}}n_y & 0 & 1 \\ \frac{1}{\sqrt{3}}n_z & -\frac{1}{\sqrt{3}}n_z & -\frac{n_x}{n_z} & -\frac{n_y}{n_z} \end{bmatrix} \tag{11a}$$

$$\boldsymbol{R}^{-1} = \begin{bmatrix} \frac{1}{2} & \frac{\sqrt{3}}{2}n_x & \frac{\sqrt{3}}{2}n_y & \frac{\sqrt{3}}{2}n_z \\ \frac{1}{2} & -\frac{\sqrt{3}}{2}n_x & -\frac{\sqrt{3}}{2}n_y & -\frac{\sqrt{3}}{2}n_z \\ 0 & (n_y^2 + n_z^2) & -n_x n_y & -n_x n_z \\ 0 & -n_x n_y & (n_x^2 + n_z^2) & -n_y n_z \end{bmatrix} \tag{11b}$$

and $\boldsymbol{\Lambda} = \mathrm{diag}(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, 0, 0)$ is the diagonal matrix of eigenvalues of $\boldsymbol{H}$.

Multiplying Eq. 9 through by $\boldsymbol{R}^{-1}$ we get

$$\boldsymbol{\beta}_t + i\kappa\boldsymbol{\Lambda}\boldsymbol{\beta} = 0, \tag{12}$$

where $\boldsymbol{\beta} = \boldsymbol{R}^{-1}\boldsymbol{\alpha}$. Because $\boldsymbol{\Lambda}$ is diagonal the solutions decouple, leading to

$$\boldsymbol{\beta} = \boldsymbol{\beta}_0\, e^{-i\kappa\lambda_i t}, \quad i = 1, 4. \tag{13}$$

Noting that $\boldsymbol{\alpha} = \boldsymbol{R}\boldsymbol{\beta}$, we can multiply this through by $\boldsymbol{R}$ to find the solutions

$$\boldsymbol{u} = \boldsymbol{\alpha}_0 \, e^{i\kappa(\boldsymbol{r}-\lambda_i t)}, \quad i = 1, 4, \tag{14}$$

for some initial amplitude $\boldsymbol{\alpha}_0$.

Therefore, under the transformation to "characteristic variables", $\boldsymbol{v} = \boldsymbol{R}^{-1}\boldsymbol{u}$, plane waves are propagated with (constant) wave–speeds defined by the non–zero eigenvalues $\pm\frac{1}{\sqrt{3}}$. Now, this is useful to us even in the time–independent case because we can identify "flow directions" with these eigenvalues relative to the direction vector of the plane wave, $\hat{n}$. If this direction is taken to be the outward normal vector of some surface, we associate outwardly directed flows through that surface with the positive eigenvalue and inwardly directed flows with the negative eigenvalue. Transformation of the (time–independent) solution vector $\boldsymbol{u} = \left[\Phi(\boldsymbol{r}), J^x(\boldsymbol{r}), J^y(\boldsymbol{r}), J^z(\boldsymbol{r})\right]^T$ to characteristic variables gives the expressions

$$\Phi^{\pm} = \frac{1}{2}\,\Phi(\boldsymbol{r}) \pm \frac{\sqrt{3}}{2}\,\hat{n}\cdot\boldsymbol{J}(\boldsymbol{r}) \tag{15}$$

that correspond to outwardly and inwardly directed flows, respectively. These equations can be used to define the "boundary" terms in Eqs. 4, as an alternative to Eqs. 2. Start by adding and subtracting a discrete version of these expressions

$$\Phi_h^b = \left(\Phi^+ + \Phi^-\right) \tag{16a}$$

$$\left(\hat{n}\cdot J_h^b\right) = \frac{1}{\sqrt{3}}\left(\Phi^+ - \Phi^-\right), \tag{16b}$$

and, for a given face j and vertex i on some cell k, find

$$\boldsymbol{a}_j\,\Phi_i^b = \boldsymbol{a}_j\left(\Phi_{j,i}^{out}\,[1+\xi_j] + \Phi_{j,i}^{in}\,[1-\xi_j]\right) \tag{17a}$$

$$\left(\boldsymbol{a}_j\cdot J_i^b\right) = \frac{|\boldsymbol{a}_j|}{\sqrt{3}}\left(\Phi_{j,i}^{out} - \Phi_{j,i}^{in}\right)[1-\xi_j], \tag{17b}$$

where the flows into and out of the face are given by

$$\Phi_{j,i}^{in} = \frac{1}{2}\,\Phi_i^{ext} - \frac{\sqrt{3}}{2}\,\hat{n}_j\cdot\boldsymbol{J}_i^{ext} \tag{17c}$$

$$\Phi_{j,i}^{out} = \frac{1}{2}\,\Phi_i + \frac{\sqrt{3}}{2}\,\hat{n}_j\cdot\boldsymbol{J}_i. \tag{17d}$$

No matter what choice is made for defining the particle flows, we order the unknowns on the mesh first by the current vector for each vertex on every cell, followed by the scalar flux for each vertex on every cell.

Then we can write the linear system in the (2×2) block form

$$\begin{bmatrix} \boldsymbol{A}_t & \frac{1}{3}\boldsymbol{A}_0 \\ -\boldsymbol{A}_0^T & \boldsymbol{A}_a \end{bmatrix} \begin{bmatrix} \boldsymbol{J} \\ \Phi \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ g \end{bmatrix}, \tag{18}$$

where the submatrices $\boldsymbol{A}_t$ and $\boldsymbol{A}_a$ are SPD. The system can also be written in the symmetric form

$$\begin{bmatrix} 3\boldsymbol{A}_t & \boldsymbol{A}_0 \\ \boldsymbol{A}_0^T & -\boldsymbol{A}_a \end{bmatrix} \begin{bmatrix} \boldsymbol{J} \\ \Phi \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ -g \end{bmatrix}, \tag{19}$$

Finally, we note that we can compute a Schur complement, $\boldsymbol{S}$, of the (2×2) block linear systems above. It is formed by block Gaussian elimination (which can be viewed as eliminating the currents in favor of the scalar fluxes) resulting in the SPD linear system

$$\boldsymbol{S}\Phi = \left[\boldsymbol{A}_a + \boldsymbol{A}_0^T(3\boldsymbol{A}_t)^{-1}\boldsymbol{A}_0\right]\Phi = g + \left[(3\boldsymbol{A}_t)^{-1}\boldsymbol{A}_0\right]f, \tag{20}$$

for the discontinuous scalar fluxes only. On tetrahedra this reduced system involves 1/4 the number of unknowns of the original full P_1 system of equations. This is potentially a tremendous savings for transport acceleration since only the scalar fluxes are needed for that purpose.

We would not form the reduced system directly because the discontinuous approximation introduced into the discretization leads to a globally coupled system. That is, both the inverse of $3\boldsymbol{A}_t$ block and, hence, the Schur complement are dense matrices. Instead we can implicitly compute the "action" of the Schur complement as needed. Iterative solution algorithms supply vectors u for which the matrix–vector product $v = Su$ is to be returned to the algorithm. The action of the Schur complement can be computed by first calculating $s = \boldsymbol{A}_0 u$ for a given vector u. Then use an inner iteration to approximately calculate $t = (3\boldsymbol{A}_t)^{-1}s$ and complete the computation by calculating $v = \boldsymbol{A}_a u + \boldsymbol{A}_0^T t$. Even though the CG algorithm is used for both SPD linear systems we have found that a combined inner–outer iteration is not as efficient as our methods for solving the full system. One way to remedy this could be to compute the Cholesky factorization of the $\boldsymbol{A}_t$ block which is first reordered for minimum fill–in. Then, instead of an inner iteration, the action the Schur complement simply requires a forward and backward substitution at every outer CG iteration. The cost of computing the factorization is amortized over the course of the CG iterations and the Schur complement matrix–vector computation will be very fast once we have stored the factorization. A drawback is that the Cholesky factorization is not necessarily efficient on parallel platforms. Furthermore, the fill–in associated with the factorization increases memory requirements. We plan to explore solution of the reduced system in the future.

2.2 Solution Methods

The discontinuous P_1 equations, Eq. 18 or Eq. 19, form an sparse, indefinite linear system, $\boldsymbol{A}x = b$. To be used as part of a DSA scheme, we must solve this system efficiently. For large problems direct methods are infeasible, so we use iterative solution techniques to which we can apply multilevel acceleration methods. Iterative solution methods also allow us to take advantage of the sparsity of the linear system by storing only the nonzeros of the matrix.

Our solution technique consists of a two–level iteration. An outer Krylov–subspace iterative method is used to solve the discontinuous P_1 equations. To compute the iterative solution efficiently, a preconditioner $\boldsymbol{M}$ is needed that can adequately alter the spectrum of the original matrix $\boldsymbol{A}$. Loosely speaking, the preconditioner will be effective if the eigenvalues of the preconditioned matrix $\boldsymbol{M}^{-1}\boldsymbol{A}$ are clustered and bounded away from the origin.[17] We have found that the convergence rate of the outer Krylov iteration improves if the condition number, measured by the ratio of the maximum to minimum singular values of $\boldsymbol{M}^{-1}\boldsymbol{A}$, is smaller than that of the original system. This is only a rough indicator of preconditioner effectiveness.[10, 18] A preconditioner for the discontinuous P_1 equations on two–dimensional Cartesian meshes was presented in Ref. 10. We have extended the method to three–dimensional unstructured tetrahedral meshes. The preconditioner uses a linear continuous finite element discretization of the diffusion equation. The iterative method used to solve the continuous diffusion equation is the second level of our two–level solution technique.

This preconditioner was shown to be effective over a wide range of problems and seems to be particularly well–suited to solving problems which are optically thick and diffusive. This is fortunate because these happen to be just the kinds of problems for which we would like to solve the P_1 equations as part of a transport acceleration algorithm. Use of the continuous diffusion equation was suggested by the observation that discontinuities in the P_1 solution disappear and approach the continuous diffusion equation solution as the problem becomes optically thick and diffusive. The preconditioner scaled very well with problem size in two dimensions but convergence rates of both the inner and outer iterations degraded for very thin or very highly skewed cells.

At every iteration, the Krylov solver supplies a residual vector r requesting that a vector $z = \boldsymbol{M}^{-1}r$ be returned to the solver. In our case, this is computed implicitly. That is, we "solve" $\boldsymbol{M}z = r$ without explicitly forming or inverting a matrix $\boldsymbol{M}$. This will made clearer if one examines our two–level approach as displayed in Algorithm 1.

Algorithm 1. Two–Level Preconditioner

```
$z \leftarrow 0$
$s \leftarrow r - \boldsymbol{A}z$
$z \leftarrow z + \omega\widetilde{\boldsymbol{A}}^{-1}s$
$s \leftarrow r - \boldsymbol{A}z$
$z \leftarrow z + \boldsymbol{C}^{-1}s$
$s \leftarrow r - \boldsymbol{A}z$
$z \leftarrow z + \omega\widetilde{\boldsymbol{A}}^{-1}s$
```

The algorithm consists of three distinct steps, giving it the character of a two–stage multigrid V–cycle. The continuous diffusion equation step, denoted by $\boldsymbol{C}^{-1}$, resides at the lowest level and is preceded and followed by damped pre– and post–smoothing iterations, where $\widetilde{\boldsymbol{A}}$ is some simple approximation to $\boldsymbol{A}$. It is this entire series of computations that is implicitly represented by the *operator* $\boldsymbol{M}^{-1}$.

The continuous diffusion *operator*, denoted by $\boldsymbol{C}^{-1}$ in Algorithm 1, is computed implicitly as well. The continuous diffusion equation is discretized with unknowns on the mesh vertices. This linear system is usually of lower dimension ($\mathbb{R}^{N_v}$, with N_v being the number of vertices in the mesh) than that of the discontinuous P_1 equations ($\mathbb{R}^{16N_c}$, where N_c is the number of cells in the mesh). The computation of $\boldsymbol{C}^{-1}$ therefore consists of three steps: a projection, a matrix inversion, and an interpolation, written symbolically as $\boldsymbol{C}^{-1} = \boldsymbol{Q}\boldsymbol{D}^{-1}\boldsymbol{P}$. The matrix $\boldsymbol{P}$ projects from from $\mathbb{R}^{16N_c}$ onto $\mathbb{R}^{N_v}$ and the matrix $\boldsymbol{Q}$ interpolates back again.

The projections and interpolation are also computed implicitly. The projections are computed as a source term for each cell–vertex centered diffusion equation by summing an appropriate combination of the discontinuous scalar flux and current residuals from all the cells surrounding a vertex. This is discussed in more detail below. The interpolations simply assign the same continuous diffusion equation solution to all the discontinuous scalar fluxes surrounding a particular vertex; the currents are left unchanged.

The matrix $\boldsymbol{D}^{-1}$ represents the inverse of the continuous diffusion equations. The solution is also computed approximately using preconditioned conjugate gradients (PCG) with simple diagonal preconditioning.

The matrix $\widetilde{\boldsymbol{A}}$ is either the block–diagonal matrix extracted from the discontinuous P_1 matrix on a cell–by–cell basis, represented by $\boldsymbol{B}$, or it is the identity $\boldsymbol{I}$. The former case is a block–Jacobi iteration and the latter is a Richardson iteration. In the block–Jacobi iteration, each block is a (16×16) matrix representing the coupling between all the unknowns (scalar fluxes and currents) on a cell. The block–Jacobi smoothing iterations are evaluated by sweeping through the mesh one cell at a time, which could potentially be very efficient in a parallel implementation. The overall preconditioner with Richardson smoothing is simpler but less effective than with block–Jacobi smoothing. In this paper we consider only $\widetilde{\boldsymbol{A}} = \boldsymbol{B}$.

Manipulation of the P_1 equations leads to a source term that represents the "correct" projection operator. We start by assuming the discontinuous unknowns in the P_1 equations are continuous at the vertices. We then write the balance equations and moment equations (in vector form) for the four vertices on some cell k. The right hand side of the P$_1$ equations is set to a "residual" vector of the discontinuous operator; referring to Algorithm 1 this vector is $s = r - \boldsymbol{A}z$, where z is the updated vector from the first block Jacobi iteration. The four moment equations are added together to find an expression for the average current vector on a cell, noting that the area vectors of a cell sum to zero and that the outwardly directed area vectors for a face shared by two cells are the negative of one another. The expression for the average current appears in the four balance equations on a cell. Inserting that expression into the balance equation for vertex j on cell k gives

$$\frac{\boldsymbol{a}_j}{27\sigma_{t,k}V_k} \cdot \left(\sum_{i=1}^{4} \boldsymbol{a}_i\,\phi_i \right) + \frac{\sigma_{a,k}V_k}{20}\left(2\phi_j + \sum_{\substack{i=1\\ i\neq j}}^{4} \phi_i \right) = s_{j,k} - \frac{\boldsymbol{a}_j}{3\sigma_{t,k}V_k} \cdot \left(\sum_{i=1}^{4} \boldsymbol{s}_{i,k} \right). \tag{21}$$

This is the continuous diffusion discretization with the right hand side being the projection of the discontinuous residual vector s, where $s_{j,k}$ is the residual in the scalar flux at vertex j in cell k and the $\boldsymbol{s}_{i,k}$ are the residuals in the currents in cell k. The continuous unknowns, ϕ_n, are given the global ordering of the mesh vertices. Every mesh vertex is shared by an arbitrary number of cells and Eq. 21 is computed for the corresponding local vertex j on each of those cells. The equations are summed over the surrounding cells, each one contributing to the coefficients of the continuous diffusion matrix, $\boldsymbol{D}$, for the row corresponding to that global vertex. The projection operation in an implementation follows from this summation.

We must use GMRES(m) if either the system or the preconditioner is not symmetric. With our definition of the projection operator and the simple interpolation method we are using, the preconditioner is not symmetric because $\boldsymbol{Q} \neq \boldsymbol{P}^T$. However, if we wish to use MINRES to solve the symmetric form of the P$_1$ equations, we can impose symmetry by *defining* $\boldsymbol{Q} = \boldsymbol{P}^T$ and we can ensure that the preconditioner is positive definite (MINRES requires the preconditioner to be SPD) by scaling the problem by some norm of the linear system.

2.3 Diffusion Synthetic Acceleration

We now present a brief description of the diffusion synthetic accelerated iterative transport solution method.

The DFEM discretization of the S$_N$ transport equation uses the same linear basis functions as the P$_1$ equations. We assume an angular quadrature $\{\hat{\Omega}_m, w_m\}$ whose weights sum to unity and consider isotropic scattering only. An inhomogeneous (isotropic) distributed source, Q_m may also be specified. The angular flux on a cell k expanded in terms of the basis functions is denoted by $\psi_{m,h}$ and the discrete problem reads as follows.

For each angle $\hat{\Omega}_m$ find an approximation to the angular flux on a cell k, $\psi_{m,h}$ satisfying

$$\hat{\Omega}_m \cdot \left(\int_{\partial T_k} \hat{n}\,\psi_m^b\,u_h\,dS - \int_{T_k} \psi_{m,h}^{\ell+1} \boldsymbol{\nabla} u_h \right) dV + \sigma_{tk} \int_{T_k} \psi_{m,h}^{\ell+1} u_h\,dV = \sigma_{sk} \sum_m w_m \int_{T_k} \psi_{m,h}^{\ell} u_h\,dV + \int_{T_k} Q_m u_h\,dV \tag{22}$$

for all trial functions u_h.

The trial functions are again just the four linear tetrahedral basis functions L_i. These expressions are computed for each vertex j on a cell k, giving four equations in four unknowns on every cell. The integrations over the boundary of the tetrahedron ∂T_k give rise to terms having the form $\left(\hat{\Omega}_m \cdot \boldsymbol{a}_j\right)\psi^b_{m,j}$. The boundary terms $\psi^b_{m,j}$ are replaced by upwinding or by the boundary conditions, specified by the function $\Gamma(\hat{\Omega})$, as follows. For a cell k and face j we set

$$\left(\hat{\Omega}_m \cdot \boldsymbol{a}_j\right)\psi^b_{m,j} = \begin{cases} \psi^{\ell+1}_{m,j,k}, & \hat{\Omega}_m \cdot \boldsymbol{a}_j > 0, \quad \forall \boldsymbol{a}_j \text{in } V \\ \psi^{ext}_{m,i,j}, & \hat{\Omega}_m \cdot \boldsymbol{a}_j < 0, \quad \boldsymbol{a}_j \text{ in } V \backslash \partial V \\ \Gamma(\hat{\Omega}_m), & \hat{\Omega}_m \cdot \boldsymbol{a}_j < 0, \quad \boldsymbol{a}_j \text{ on } \partial V, \end{cases} \tag{23}$$

where the "external" flux $\psi^{ext}_{i,j}$ is the value of the angular flux across the face j from vertex i in the neighboring cell that shares face j with cell k at iteration $\ell+1$. Hence, if $\boldsymbol{a}_j$ is on the boundary of the problem domain V, then the boundary condition is used to define the incoming angular flux; otherwise the internal or external values angular fluxes are used depending on the orientation of the cell face with respect to the quadrature direction.

For each quadrature angle $\hat{\Omega}_m$, the transport discretization computes a the angular flux, $\psi_{m,i,j}$ for each vertex j on every cell k. We can write the source iteration in matrix notation as

$$\boldsymbol{T}_m \psi^{\ell+1}_m = \sigma_s \, \boldsymbol{S}\phi^\ell + Q_m, \quad \phi^\ell = \sum_m w_m \psi^\ell_m, \tag{24}$$

where the correspondence to Eq. 22 is obvious. With DSA, the source iteration is modified as follows.

$$\psi^{\ell+1/2}_m = \sigma_s \boldsymbol{T}^{-1}_m \boldsymbol{S}\phi^\ell + \boldsymbol{T}^{-1}_m Q_m \tag{25a}$$

$$\phi^{\ell+1/2} = \sum_m w_m \psi^{\ell+1/2}_m \tag{25b}$$

$$\epsilon^{\ell+1/2} = \sigma_s \, \boldsymbol{X}^T \boldsymbol{A}^{-1} \boldsymbol{X}\left(\phi^{\ell+1/2} - \phi^\ell\right) \tag{25c}$$

$$\phi^{\ell+1} = \phi^{\ell+1/2} + \epsilon^{\ell+1/2} \tag{25d}$$

where $\boldsymbol{A}$ represents the P_1 equations, Eqs. 4, with the upwinding equations, Eqs. 6 or 17. Note that the scalar flux transport residuals only contribute a source to the balance equations of the P_1 equations, the moment equations sources being zero. Similarly, the transport scalar fluxes are corrected only by the P_1 equation solution for the scalar fluxes. The matrix $\boldsymbol{X}$ accounts for these "projections".

2.4 Improving the Efficiency of the Iterative Solution

Krylov iterative methods are used to compute the solution of the discontinuous P_1 equations. We will now suggest two simple ways in which we can reduce the number of Krylov iterations to improve the overall efficiency of the accelerated source iterations.

One way to improve efficiency might be to accept the Krylov subspace GMRES (or MINRES) solution of the P_1 equations after a fixed number of iterations. This can be done as long as the Krylov method converges monotonically in whatever norm is being used to monitor convergence. For general use, though, we consider this approach to be unacceptable because the minimum number of iterations that can be taken without adversely affecting source iteration convergence is not known ahead of time.

Another way would be to relax the convergence tolerance. We found that we could reduce the total number of iterations spent solving the P_1 equations in the DSA algorithm without affecting the source iteration convergence by varying the convergence tolerance during the course of the S_N iterations as follows. Let E^ℓ be the discrete L-2 norm of the relative change in cell–average scalar fluxes between two successive iterations. Then we set the tolerance τ^ℓ for the GMRES (or MINRES) solver to be proportional to E^ℓ:

$$\tau^\ell = \begin{cases} \frac{1}{10}\max(\frac{1}{10}, \epsilon) & \text{if } \ell = 1 \\ \frac{1}{10}\max(\min(E^\ell, \frac{1}{10}), \epsilon) & \text{otherwise.} \end{cases}$$

Here ϵ is the tolerance of S_N source iteration convergence. The factors of 1/10 apprearing here are chosen conservatively. We found that this simple heuristic worked very well.

We can also improve the overall efficiency by altering the convergence tolerance of the PCG iterations in the preconditioner for the P_1 equation. We use an approach for improving the solution efficiency of nested inner–outer Krylov methods presented in Ref. 19. This approach can be roughly described by noting that at any particular iteration a Krylov subspace solver constructs a solution based on the solutions from previous iterations to that point. It is therefore logical that the inner iterations should be computed with a strict convergence tolerance in the early stages of the outer iterative solution and the tolerance can be relaxed as the outer iteration proceeds. This is somewhat contrary to intuition and the reasons behind this observation are as yet not fully understood mathematically.[19]

One way to do this would be to make the inner iteration tolerance proportional to the inverse of the norm of the residual of the outer iteration. The discontinuous P_1 equations are solved with just such an inner–outer Krylov method, the outer method in this case being the GMRES (or MINRES) iteration and the inner method being the PCG iteration for the continuous diffusion equation preconditioner. This is opposite to the strategy we used for the S_N iterations and the fully consistent DSA method and contrary to intuition. Assume that at some iteration n, the current outer iteration has a residual norm r^n. Then we set the inner PCG convergence tolerance to

$$\gamma = \begin{cases} \frac{1}{10}\tau & \text{if } r^n = 0 \\ \frac{1}{10}\min(1, \tau/\min(r^n, 1)) & \text{otherwise} \end{cases}$$

where τ is the tolerance for the outer iteration. Again, we consider the factors of 1/10 to be a conservative choice. In the problems we tried this approach also worked very well, reducing the number of PCG iterations without affecting convergence of the GMRES (or MINRES) iteration. The anticipated improvements in overall efficiency with this approach are limited, however, because the PCG iterations are already very fast.

3. FOURIER ANALYSIS

In this section we present a three–dimensional Fourier analysis on tetrahedra. We can use these results to predict how effective DSA methods in accelerating source iteration convergence on unstructured tetrahedral grids.

The underlying Fourier ansatz is made on a three–dimensional Cartesian grid, the basic element of which is a box of dimension $\Delta x \times \Delta y \times \Delta z$. The box is divided into six tetrahedra whose edges must line up when the basic elements are "translated" in order to "tile" the Cartesian grid with tetrahedra. The minimum number of tetrahedra that satisfy this requirement is six. This basic element is illustrated in Fig. 2.

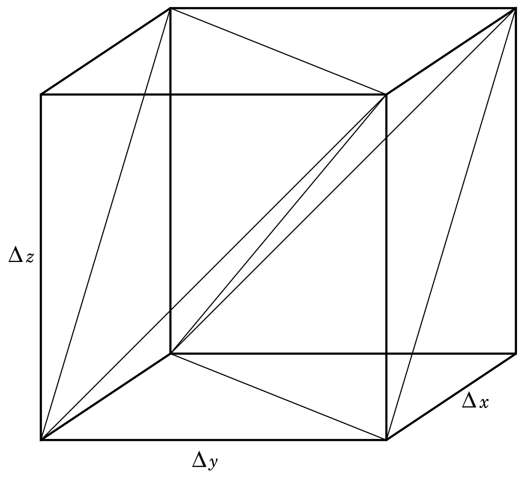

Figure 2. The basic element for the Fourier analysis divided into six tetrahedra of equal volume.

The tetrahedral cells on the basic element are numbered from $k = 1, \ldots, 6$, each of the four vertices are locally ordered within each tetrahedron from $j = 1, \ldots, 4$, and the four quantities at each vertex in every cell in the basic element are given globally ordered as $i = 4(k-1) + j$. The Fourier ansatz assumes the error modes in the quantities $\Phi_i, J_i^x, J_i^y, J_i^z$, $i = 1, \ldots, 24$, can be represented in the discrete form

$$\begin{aligned}
\Phi_{j,k}(\boldsymbol{r}) &= \hat{\Phi}_i e^{\,\mathrm{i}\left(\bar{\Lambda}\cdot\boldsymbol{r}\right)} \\
J_{j,k}^x(\boldsymbol{r}) &= \hat{J}_i^x e^{\,\mathrm{i}\left(\bar{\Lambda}\cdot\boldsymbol{r}\right)} \\
J_{j,k}^y(\boldsymbol{r}) &= \hat{J}_i^y e^{\,\mathrm{i}\left(\bar{\Lambda}\cdot\boldsymbol{r}\right)} \\
J_{j,k}^z(\boldsymbol{r}) &= \hat{J}_i^z e^{\,\mathrm{i}\left(\bar{\Lambda}\cdot\boldsymbol{r}\right)}.
\end{aligned}$$

Here, $\bar{\Lambda} = \left[\lambda_x, \lambda_y, \lambda_z\right]^T$ is the vector of Fourier wave numbers. The terms for cells in the basic element that have faces on the boundary and which exist outside the basic element (the "external" terms Φ_k^{ext} and $\boldsymbol{J}_k^{ext}$ in Eqs. 6 or 17) are defined in terms of quantities interior to the basic element that are "translated" by the width of the basic element through the Fourier ansatz.

For example, consider the quantities that happen to have global indices $i = 14, 15, 16$ and suppose they are located in a cell on the three vertices of a cell face that is on the left side of the basic element. Also suppose that another cell outside the basic element has one of its faces on the right side of the basic element as illustrated in Fig. 3. The cell of interest inside the basic element has been shaded. Remember that although the locations of the quantities are shown to lie just inside the vertices for illustration they are actually defined mathematically as limiting values of the unknowns at the vertices. The boundary terms that couple cell $k = 4$ to those in the adjacent cell which would be sharing the face on the right side of the basic element are denoted by a, b and c in the figure. The Fourier ansatz for the quantities outside the basic element at those points, coming from the "exterior" values in the upwinding equations, "look like" the quantities at $i = 14, 15, 16$ that are inside the basic element. We choose the "origin" of the basic element to be the vertex at the rear lower left corner of the box in Fig. 3. Then, when the equations for cell $k = 4$ are constructed, the values for the terms in the cells adjacent to the right face of the cell at the points a, b and c are assigned the "translated" values of the basic quantities as follows:

$$\begin{aligned}
&\text{at point a:} \quad \hat{\Phi}_{14}\, e^{\,\mathrm{i}\,\lambda_y \Delta y}, \quad \hat{\boldsymbol{J}}_{14}\, e^{\,\mathrm{i}\,\lambda_y \Delta y}; \\
&\text{at point b:} \quad \hat{\Phi}_{15}\, e^{\,\mathrm{i}\,\lambda_y \Delta y}, \quad \hat{\boldsymbol{J}}_{15}\, e^{\,\mathrm{i}\,\lambda_y \Delta y}; \\
&\text{at point c:} \quad \hat{\Phi}_{16}\, e^{\,\mathrm{i}\,\lambda_y \Delta y}, \quad \hat{\boldsymbol{J}}_{16}\, e^{\,\mathrm{i}\,\lambda_y \Delta y}.
\end{aligned}$$

Similar assignments are made for all the cells whose equations are coupled to faces adjoining the exterior sides of the basic element.

Using the symbolic algebra program MAPLE, Equations 4, together with the upwinding equations, Eqs. 6 or 17, are written for all cells and all vertices on the basic element. The Fourier ansatz, including the translated boundary terms from quantities outside of the basic element, is made in the 96 equations on

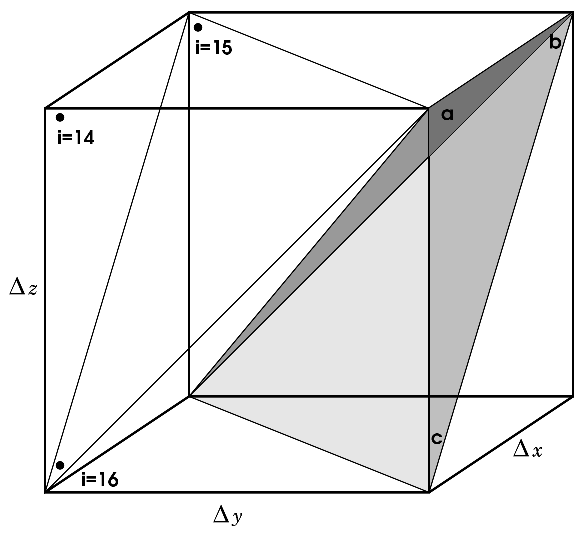

Figure 3. An example illustrating the Fourier ansatz. The cell $k = 4$ is shaded. The points a, b and c lie in the cell *outside* the basic element that is adjacent to the shaded cell. The quantities at those points couple cell $k = 4$ to the adjacent cell through the discontinuous "upwind" approximation. They are represented in the Fourier analysis by the quantities for a cell that lies *inside* the basic element. In this case, the quantities at the vertices labeled $i = 14, 15$, and 16, are "translated" by the Fourier ansatz $e^{\mathrm{i}(\bar{\Lambda}\cdot\boldsymbol{r})}$. The "origin" is assumed to be at the rear lower left corner of the box.

the basic element. The (96×96) matrix is denoted by $\widetilde{\boldsymbol{A}}$. It is computed in terms of the basic element thickness $\Delta x, \Delta y$, and Δz, the Fourier wave numbers λ_x, λ_y and λ_z, and the *constant* material properties, the scattering ratio c and total cross section σ_t.

For every discrete ordinate m, a Fourier ansatz of the form

$$\psi_{m,j,k}(\boldsymbol{r}) = \hat{\psi}_{m,i} e^{\mathrm{i}(\bar{\Lambda}\cdot\boldsymbol{r})}$$

is also made for the errors in the angular fluxes $\psi_{m,j,k}$ of the transport equation Eq. 22. The basic element quantities are $\hat{\psi}_{m,i},\ i = 1, \ldots, 24$. The upwinding Eqs. 23 introduce quantities from outside the basic element which are represented by "translated" quantities from inside the basic element. We use the same global ordering and the same "origin" for the basic element to be consistent the Fourier representation of the P$_1$ equations. In matrix notation, the transport equation can be written for source iteration index ℓ as

$$\widehat{\boldsymbol{T}}_m \hat{\psi}_m^{\ell+1} = \sigma_s \sum_m w_m \widehat{\boldsymbol{S}} \hat{\psi}_m^{\ell}, \tag{26}$$

where $\hat{\psi}_m$ represents the vector of Fourier quantities.

For unaccelerated transport, the (24×24) matrix

$$\widehat{\boldsymbol{F}} = \sum_m w_m \widehat{\boldsymbol{T}}_m^{-1} \widehat{\boldsymbol{S}}, \tag{27}$$

can then also be computed in terms of the Fourier wave number, the basic element thickness, and the material properties using symbolic expressions from MAPLE. The maximum eigenvalue of this matrix is the unaccelerated spectral radius. We have verified that it is equal to $c = \sigma_s/\sigma_t$.

Now, the DSA algorithm in Eqs. 25a leads to the (24×24) matrix ($\widehat{\boldsymbol{I}}$ is the identity)

$$\widehat{\boldsymbol{G}} = \Big[\widehat{\boldsymbol{F}} + \sigma_s\, \widehat{\boldsymbol{X}}^T \widehat{\boldsymbol{A}}^{-1} \widehat{\boldsymbol{X}} \big(\widehat{\boldsymbol{F}} - \widehat{\boldsymbol{I}}\big)\Big], \tag{28}$$

whose maximum eigenvalue is the spectral radius of the accelerated transport solution. The matrix $\widehat{\boldsymbol{X}}$ is a discrete "projection" matrix of dimensions (96×24).

The matrix $\widehat{\boldsymbol{G}}$ is computed with MAPLE by combining symbolic expressions for $\widehat{\boldsymbol{A}}^{-1}$, $\widehat{\boldsymbol{F}}$, and $\widehat{\boldsymbol{X}}$. It is evaluated for fixed parameters and maximized over all frequencies using a Nelder–Mead simplex algorithm with quadratic surface fitting near suspected maxima. We found this algorithm to be essential in searching for the maximum over the three–dimensional space of wave numbers.

Note that we assume the discontinuous P_1 equations, represented by $\widetilde{\boldsymbol{A}}$ in the Fourier analysis, are inverted exactly. In an actual implementation, the matrix $\boldsymbol{A}$ is only approximately inverted. This means that the actual source iteration convergence could be delayed and the measured spectral radius could be higher than Fourier analysis predicts.

We end this section by giving an expression for the analytical spectral radius of the DSA algorithm, ρ_0:[4, 6]

$$\rho_0 = \max_{\lambda_x, \lambda_y, \lambda_z} \left| \omega + \frac{c}{\alpha}(\omega - 1) \right|, \tag{29a}$$

$$\text{where} \qquad \omega = \frac{c}{4\pi} \int_{4\Pi} d\Omega \Big[1 + \mathrm{i}\,(\bar{\Lambda} \cdot \hat{\Omega})\Big]^{-1} \tag{29b}$$

$$\text{and} \qquad \alpha = \frac{1}{3}(\bar{\Lambda} \cdot \bar{\Lambda}) + (1 - c). \tag{29c}$$

We can evaluate the integral in Eqs. 29 with an adaptive numerical integration. The analytical spectral radius corresponding to a specified discrete ordinates quadrature, denoted by $\tilde{\rho}_0$, is calculated by evaluating the integral with the quadrature. In either case, the Nelder–Mead simplex algorithm is again used to search over the space of Fourier wave numbers on $[0, 2\pi)$.

4. NUMERICAL RESULTS

In this section we will investigate the effectiveness and efficiency of the fully consistent DSA (FCDSA) scheme using our solution method for the P_1 equations. Numerical results are computed using our implementation code, ATTILAV2, as described in Wareing, et al. (2001). Theoretical predictions of the spectral radius are computed using the results of the Fourier analysis of the previous section.

In the results that follow, Fourier analysis predictions and measurements of the spectral radius will be given for the FCDSA scheme. They will be compared to those of the partially consistent discretization M4S DSA scheme.[8] In all the computations reported here, we use Algorithm 1 with $\widetilde{\boldsymbol{A}} = \boldsymbol{B}$ to solve Eq. 19 using GMRES. Note that same algorithm can be used to accelerate the M4S DSA equations.

ATTILAV2 currently uses the partially consistent WLA DSA scheme.[11] Although we have not performed a Fourier analysis of this method, we will report measured values of the spectral radius since WLA DSA is available in the implementation code.

We consider only isotropic scattering in which case we expect the fully consistent method to be stable for all cell widths and cell aspect ratios.

The first set of results, shown in Table 4., compares the spectral radius predicted by Fourier analysis to the measured spectral radius from ATTILAV2 transport code for a scattering ratio $c = 0.9999$ and total cross section $\sigma_t = 3.5\text{cm}^{-1}$. The results are listed for representative cell widths (in terms of the size of the basic element "box") in order of decreasing aspect ratio. The aspect ratio measure is computed as the three times the ratio of the inscribed to circumscribed spheres. It is less than or equal to one, attaining its maximum for a tetrahedron with edges of equal length and approaching zero as the tetrahedra become more distorted.[20] The minimum aspect ratio, α_{min}, of the six tetrahedra in the basic element is listed in the table.

The spectral radius measurements were computed on a fixed $(8 \times 8 \times 8)$ grid of boxes, each divided into six tetrahedra, for a total of 3072 cells in the problem. The outer problem dimensions were varied to alter the aspect ratio of the tetrahedral cells; for example, to have a basic element of size $(2.0\,\text{cm} \times 1.0\,\text{cm} \times 5.0\,\text{cm})$, the problem domain is $x \in [0, 16\,\text{cm}], y \in [0, 8\,\text{cm}], z \in [0, 40\,\text{cm}]$. Boundary conditions on the bottom, left, and back faces of the problem are reflective, the others are vacuum. Sources are set to zero and the angular fluxes are initialized randomly and the scalar fluxes are normalized to the total scalar flux in the problem after each iteration. We use an S_4 triangular, Chebyshev-Legendre (TCL) quadrature and a relative L-2 norm convergence criterion of 10^{-4} for the inner iterative solution of the P_1 equations. We use a fixed tolerance of 10^{-5} for the PCG iterations on the continuous diffusion equations in Algorithm 1. The spectral radius is measured as the ratio of the change in the discrete L2 norm between successive S_N iterations, reported at the end of 100 iterations. The WLA spectral radius is reported after 300 iterations.

These results indicate that the fully consistent DSA method is stable and effective whereas the partially consistent method of Adams and Martin can be unstable when the aspect ratio is small. The measured spectral radius is expected to be less than the Fourier analysis predicts because of leakage from the vacuum boundaries. The M4S and WLA methods show a strong dependence on cell aspect ratio and cell thickness. The M4S method becomes unstable as the aspect ratio decreases. This is an unexpected result since the initial verification of the method in one– and two–dimensions did not indicate any instability.[8] The spectral radius of the WLA method increases as the aspect ratio decreases. However, we want to emphasize that this is for a scattering ratio very close to unity and in general the method is very effective when the scattering ratio is not so close to one. Nonetheless, in applications any degradation in effectiveness of the WLA DSA scheme is often compensated by its computational efficiency. The fully consistent method appears to be unaffected by cell aspect ratio.

We note that the spectral radius measurements were insensitive to the tolerance used to terminate the iterative solution of the P_1 equations. This means it is possible to relax this tolerance during the early stages of the S_N source iterations and tighten it as the iterations proceed without affecting the source iteration convergence.

Table I. Fourier analysis and computationally measured spectral radius for DSA–accelerated transport using the TCL S_4 quadrature. The spectral radius is tabulated as a function of the minimum tetrahedral cell aspect ratio in the basic element, α_{min}.

				FCDSA		M4S		WLA	
α_{min}	Δx	Δy	Δz	Analysis	Measured	Analysis	Measured	Analysis	Measured
0.632	1.0	1.0	1.0	0.2215	0.2165	0.5436	0.5064	0.6729	0.6550
0.562	2.0	2.0	3.0	0.1706	0.1688	0.5851	0.5661	0.8530	0.8402
0.487	1.0	1.0	2.0	0.2062	0.2065	0.6420	0.6145	0.7994	0.7846
0.421	2.0	2.0	5.0	0.1714	0.1672	0.7043	0.6778	0.9051	0.8927
0.370	2.0	1.0	3.0	0.2066	0.2036	0.9586	0.9174	0.8574	0.8429
0.327	3.0	1.0	3.0	0.2008	0.1982	1.0783	1.0339	0.8588	0.8485
0.256	2.0	1.0	5.0	0.2123	0.2094	1.1173	1.0670	0.9096	0.8969
0.170	2.0	1.0	8.0	0.2054	0.2024	1.2254	1.1680	0.9402	0.9298
0.116	8.0	1.0	10.0	0.1482	0.1461	1.4755	1.4425	0.9540	0.9469

The next set of results are shown in Fig. 4. We compare the measured spectral radius and the Fourier analysis for the three DSA schemes as before. In our measurements the scattering ratio is again taken to be $c = 0.9999$. This time, however, the total cross section is varied logarithmically from 2^{-5}cm^{-1} to 2^{10}cm^{-1} and the basic element size is fixed ($2.0\,\text{cm} \times 1.0\,\text{cm} \times 8.0\,\text{cm}$) for which the minimum cell aspect ratio is 0.170. The measured spectral radius was computed on a fixed $(6 \times 6 \times 6)$ grid of boxes, each divided into six tetrahedra, for a total of 1296 cells, and the problem domain is fixed at $x \in [0, 12\text{ cm}], y \in [0, 6\text{ cm}], z \in [0, 48\text{ cm}]$. The boundary conditions again consistent of three reflective faces and three vacuum faces.

The spectral radius ρ is expected to approach the discrete ordinates analytical value $\tilde{\rho}_0$ in the limit of vanishing cell thickness and should approach zero for optically thick cells.[4, 6] The discrete ordinates analytical spectral radius for the S_4 TCL quadrature, $\tilde{\rho}_0 = 0.2542$, is shown in the figure for comparison. For thin cells, the Fourier analysis indeed approaches the analytical value. The measured results drop off from the exact result because of leakage as the problem becomes extremely thin. The fully consistent scheme is stable and effective for all cell optical thickness. The M4S method becomes unstable for intermediate optical thickness. While both the FCDSA and M4S methods become increasingly effective as the problem becomes thick and diffusive, the effectiveness of the WLA method degrades with increasing optical thickness.

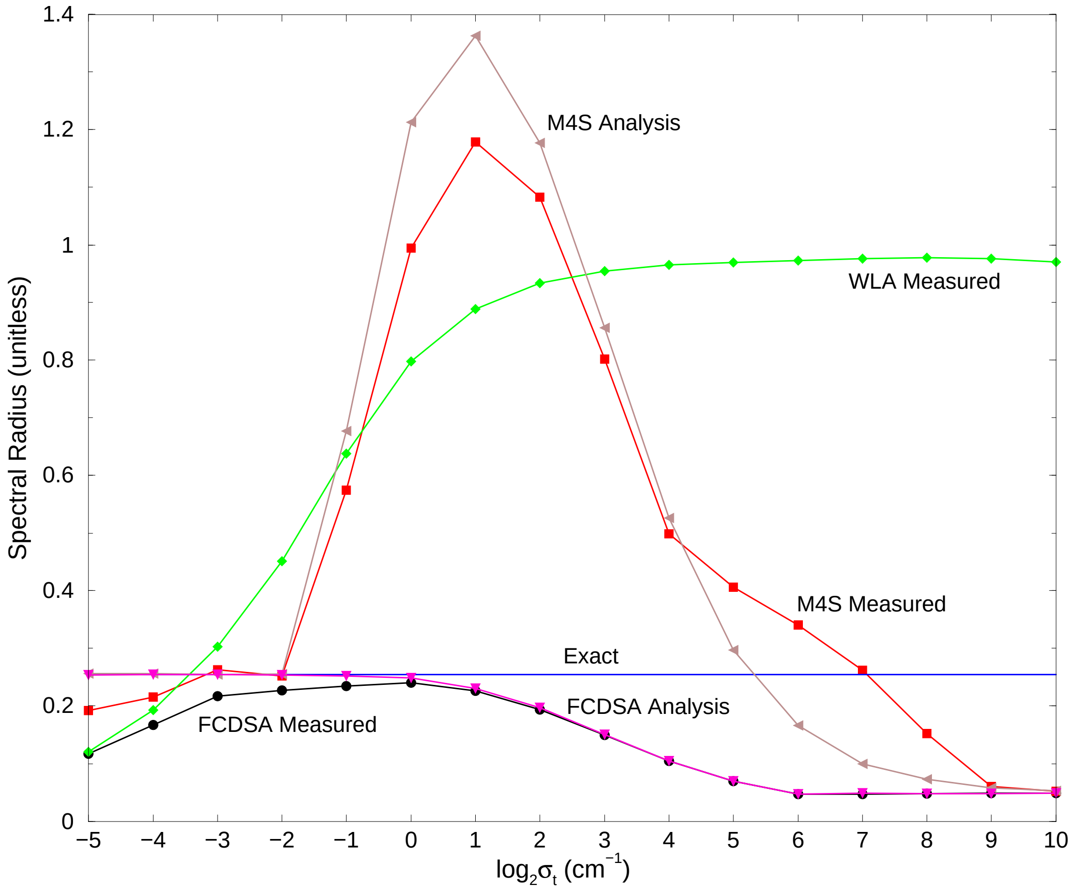

Figure 4. Fourier analysis and computationally measured spectral radius for DSA–accelerated transport for the S_4 TCL quadrature. The spectral radius is shown as a function of total cross section.

The last set of results compares the computational effort needed to compute an S_N solution with the FCDSA method to that of the WLA DSA scheme. For this problem, we solve a one–group, steady state, oil well logging tool problem, a 139.7 cm tall half cylinder of radius 60 cm, modeled with an unstructured mesh of 43,012 cells.[11] There are two He-3 detectors and a unit source of neutrons inside the problem. The minimum aspect ratio for the mesh is 0.1234 while the maximum is 0.9996. The material cross-sections are given in Table (4.). Isotropic scattering is assumed for all materials.

Table II. Neutron cross sections for the oil well logging tool problem.

	Cross sections (cm^{-1})	
Material	Total	Scattering
Limestone	$8.79672 \cdot 10^{-1}$	$8.70516 \cdot 10^{-1}$
Iron	$1.16776 \cdot 10^{0}$	$9.66125 \cdot 10^{-1}$
Water	$3.13459 \cdot 10^{0}$	$3.11519 \cdot 10^{0}$
He-3	$4.94621 \cdot 10^{-1}$	$1.00243 \cdot 10^{-4}$

The number of floating point operations (FLOP) needed to compute the S_N solution to a relative convergence criteria of 10^{-5} in the scalar fluxes on an SGI Origin 2000 processor are shown in Table. 4. for a range of TCL quadrature orders N. The FLOP count is a measure of computational effort not affected by memory access issues and is independent of data layout or other system resource use.

The highly scattering, diffusive water–containing regions of the problem brings the unaccelerated spectral radius to approximately 0.9916. This is the measured spectral radius for the S_4 quadrature which used $266.4 \cdot 10^9$ FLOP to converge in 2234 iterations. The FLOP count is roughly proportional to the solution time; for comparison, the unaccelerated solution was computed in 215 CPU minutes. Noting that this is only a one group problem (the cross sections correspond to the lowest neutron energy group in a 47 group library), it is clear that DSA is necessary for practical applications.

Because of the low minimum cell aspect ratio, the M4S DSA method is unstable. For example, after the third iteration with the S_4 quadrature the spectral radius is 1.5136 and the method never recovers. This observation is independent of quadrature order.

The table shows that the WLA method is very efficient with each DSA step taking a very small fraction of the total solution time whereas the FCDSA algorithm takes a majority of the computation time. However, because it is so effective in reducing the spectral radius – FCDSA converged in 13 iterations and WLA in 102 – the overall computation time can be less than that of the WLA method when the quadrature order is large enough.

Table III. Floating point operation counts (in billions) for the oil well logging tool problem for various S_N quadrature orders N. The total needed to compute the S_N solution is tabulated along with the counts spent in just the DSA algorithm (percent of total is in parentheses).

	FCDSA		WLA	
N	Total	DSA	Total	DSA
4	111.4	110.0 (98.7%)	13.7	1.683 (12.3%)
8	115.4	110.6 (95.8%)	41.5	1.684 (4.06%)
12	120.6	110.6 (91.7%)	85.0	1.684 (1.98%)
16	128.2	111.1 (86.7%)	144.5	1.684 (1.17%)
20	137.1	111.1 (81.0%)	219.8	1.684 (0.77%)
24	148.0	111.1 (75.1%)	310.9	1.684 (0.54%)

5. CONCLUSIONS

We have found that our fully consistent DSA scheme for DFEM discretizations of the S_N equations based on an analogous DFEM discretization of the P_1 equations is stable and very effective over a wide range of cell shapes, dimensions and optical thicknesses. For problems with a low aspect ratio, we found that the partially consistent M4S DSA scheme can be unstable on unstructured grids while the effectiveness of the WLA DSA method degrades for optically thick, diffusive problems. Our results were verified both analytically using our three–dimensional Fourier analysis and numerically with the implementation code ATTILAV2. Degradation of the WLA method in very diffusive and thick problems was expected. On the other hand, the instability of the M4S method was not anticipated in view of the previous work in one and two dimensions showing stability. The M4S DSA scheme should probably not be implemented for general use because it is not known in advance if the the method will be stable for any given problem.

We expected that the increased complexity of the fully consistent DSA scheme could make it impractical for some problems. But it could be more computationally efficient than the partially consistent WLA DSA method under certain circumstances. For example, a problem may require a high quadrature order because it contains both streaming regions and diffusive regions. We have seen that the fully consistent method could be more efficient in that case. The most promising application for the fully consistent method is probably thermal radiative transfer in the stellar regime, where the scattering ratio is often extremely close to one.

So far we have implemented our methods in serial codes only. Our conclusions could change when we extend our method to parallel platforms. For instance, solving the symmetric form of the discontinuous P_1 equations with MINRES, even with the less efficient symmetrized preconditioner, could outperform GMRES(m) in parallel. Or, possibly, the computational efficiency of direct methods might be competetive with iterative methods in parallel implementations. In that case, concern over the increased memory requirements associated with fill–in of the matrix during factorization is alleviated to some extent on a distributed memory platform. A discussion of the issues involved in choosing a solution method on parallel machines can be found in Ref. 21.

Finally, we plan to pursue methods for solving the reduced Schur complement system for use in DSA applications. Solving the reduced system could make the fully consistent DSA method more competitive. There are applications other than DSA that require the full discontinuous P_1 solution. In that case an efficient solution of the reduced system could be used as part of a very effective preconditioner for the full system. Furthermore, we may be able to compute approximate Schur complement systems that could serve as stable and effective DSA schemes. We are currently exploring these possibilities.

Acknowledgements

The authors would like to thank Michele Benzi for his advice and perspective on the linear algebraic aspects and Krylov solution methods used in this work. This work was performed under the auspices of the U.S. Department of Energy at the Los Alamos National Laboratory.

References

1. T. A. Wareing, J. M. McGhee, J. E. Morel, and S. D. Pautz, “Discontinuous Finite Element S_n Methods on Three–Dimensional Unstructured Grids,” *Nucl. Sci. and Engr.*, **138**, pp. 1–13 (2001).

2. E. M. Gelbard and L. A. Hageman, “The Synthetic Method as Applied to the S_n Equations,” *Nucl. Sci. and Engr.*, **37**, 288 (1969).

3. R. E. Alcouffe, “Diffusion Synthetic Acceleration Methods for Diamond–Differenced Discrete–Ordinates Equations,” *Nucl. Sci. and Engr.*, **64**, 344 (1977).

4. E. W. Larsen, “Unconditionally Stable Diffusion–Synthetic Acceleration Methods for Slab Geometry Discrete Ordinates Equations. Part I: Theory,” *Nucl. Sci. and Engr.*, **82**, pp. 47–63 (1982).

5. D. R. McCoy and E. W. Larsen, “Unconditionally Stable Diffusion–Synthetic Acceleration Methods for Slab Geometry Discrete Ordinates Equations. Part II: Numerical Results,” *Nucl. Sci. and Engr.*, **82**, 64 (1982).

6. M. L. Adams and T. A. Wareing, “Diffusion-Synthetic Acceleration Given Anisotropic Scattering, General Quadratures, and Multidimensions,” *Trans. of the Am. Nucl. Soc.*, **68**, 203 (1993).

7. T. A. Wareing, E. W. Larsen, and M. L. Adams, “Diffusion Accelerated Discontinuous Finite Element Schemes for the Sn Equations in Slab and X–Y Geometries,” in **Proc. International Topical Meeting on Adavances in Mathematics, Computations, Reactor Physics**, Vol. 3, Pittsburgh, Pennsylvannia, 11.1 2.1 (1991).

8. M. L. Adams and W. R. Martin, “Diffusion Synthetic Acceleration of Discontinuous Finite Element Transport Iterations,” *Nucl. Sci. and Engr.*, **111**, 145 (1992).

9. J. E. Morel, J. E. Dendy, and T. A. Wareing, “Diffusion–Accelerated Solution of the Two–Dimensional S_n Equations with Bilinear–Discontinuous Differencing,” *Nucl. Sci. and Engr.*, **115**, 304 (1993).

10. J. S. Warsa, T. A. Wareing, and J. E. Morel, “Solution of the Discontinuous P_1 Equations in Two–Dimensional Cartesian Geometry with Two–Level Preconditioning,” *SIAM J. Sci. Comp.* (2000). submitted.

11. T. A. Wareing, J. M. McGhee, and J. E. Morel, “ATTILA: A Three–Dimensional, Unstructured Tetrahedral Mesh Discrete Ordinates Transport Code,” *Trans. of the Am. Nucl. Soc.*, **75**, pp. 146–147 (1996).

12. D. N. Arnold, F. Brezzi, B. Cockburn, and D. Marini, “Discontinuous Galerkin Methods for Elliptic Problems,” in **Discontinuous Galerkin Methods** (B. Cockburn, G. E. Karniadakis, and C.-W. Shu, Eds.), 89, Berlin Heidelberg: Springer–Verlag (2000).

13. B. Cockburn, G. E. Karniadakis, and C.-W. Shu, Eds., **Discontinuous Galerkin Methods**. Berlin Heidelberg: Springer–Verlag (2000).

14. O. C. Zienkiewicz and R. L. Taylor, **The Finite Element Method**, Vol. 1. London: McGraw–Hill, Fourth Edition (1994).

15. R. J. LeVeque, **Numerical Methods for Conservation Laws**. Basel: Birkhauser Verlag (1990).

16. C. Hirsch, **Numerical Computation of Internal and External Flows**, Vol. 1 *Fundamentals of Numerical Discretization.* New York: Wiley (1988).

17. S. L. Campbell, I. C. F. Ipsen, C. T. Kelley, and C. D. Meyer, "GMRES and the Minimal Polynomial," *BIT*, **36**, 664 (1996).

18. N. M. Nachtigal, S. C. Reddy, and L. N. Trefethen, "How Fast Are Nonsymmetric Matrix Iterations?," *SIAM J. Matrix Anal. and Apps.*, **13**, 778 (1992).

19. A. Bouras and V. Frayssé, "A Relaxation Strategy for Inexact Matrix-Vector Products for Krylov Methods," CERFACS TR/PA/00/15, European Centre for Research and Advanced Training in Scientific Computation (Sep 2000). (Submitted to SIAM J. Matrix Anal. and Apps.).

20. A. Liu and B. Joe, "Relationship Between Tetrahedron Shape Measures," *BIT*, **34**, 268 (1994).

21. I. S. Duff and H. A. van der Vorst, "Developments and Trends in the Parallel Solution for Linear Systems," CERFACS TR/PA/99/10, European Centre for Research and Advanced Training in Scientific Computation (Apr 1999).